DUAL-FRAME USER INTERFACE ON GENERIC CLIENT SOFTWARE

Bryan G. Yamamoto

CROSS-REFERENCE TO COMPUTER PROGRAM LISTING APPENDIX

[0001] A computer program listing appendix, incorporated herein by reference, is submitted as part of this disclosure. The computer program listing appendix is stored under the file name: "toc.js" residing on one compact disk.

[0002] A portion of the disclosure of this patent document contains material which is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure, as it appears in the Patent and Trademark Office patent files or records, but other wise reserves all copyright rights whatsoever.

FIELD OF THE INVENTION

[0003] The present invention relates network based applications. More specifically, the present invention relates to data viewing on generic client software, such as web browsers.

BACKGROUND OF THE INVENTION

[0004] Network applications typically refers to computer applications on a first computer that interact with other computer applications running on a second computer. Network applications typically operate in a client-server paradigm or a peer-to-peer paradigm. Generally, one skilled in the art can easily adapt a network application to use either the client-server paradigm or the peer-to-peer paradigm. Thus, for clarity and conciseness, only client-server network applications are described herein.

[0005] Fig. 1 shows a typical layout for several computers. Specifically, Fig. 1 includes a server computer 110 with server software 115, a local area network 120, a client computer 140 with custom client software 145, a wide area network 150, and a client computer 160 with generic client software 165.

[0006] Server computer 110 is coupled to client computer 140 through local area network 120. Common types of local area networks include Ethernet, token ring, and FDDI. Local area network 120 is coupled to wide area network 150, which can be for example the internet. Typically, a router (not shown) is used to couple local area network 120 to wide area network 150. Client computer 160 is coupled to wide area network 150, typically through a modem (not shown) or another local area network (not shown). The data bandwidth of local area network 120 is typically between 10 and 100 megabits per seconds. However, the wide area network connection between client computer 160 and server computer 110 may be much slower such as 56 kilobits per second.

[0007] Server computer 110 typically contains data records, such as electronic mail (email) messages, data files, and electronic forum messages (e.g., newsgroup postings), which are accessible to users on client computers. Generally, server software 115 runs on server computer 110 to allow client software to access the desired data records. On local computers, i.e., computers on the same local area network such as client computer 140, custom client software, such as custom client software 145, is installed to communicate and transfer the desired data records with server software 115. For example, if the data records are email, server software 115 is a mail server such as Microsoft Exchange™ and custom client software 145 is a mail client such as Microsoft Outlook™. Because of the fast connection between local computers provided by local area network 120 and the customized

features of custom client software 145, a user of client computer 140 is provided many features and benefits as described below.

[0008] Non-local client computers do not necessarily have custom client software 145. However, most computers are now configured with generic client software such as web browser (e.g., Microsoft Internet Explorer™ or Netscape Navigator™). Generic client software communicates with server software using industry standard protocols such as hypertext markup language (HTML) and JavaScript™. Server software can control generic client software to simulate some of the features of custom client software.

[0009] Fig. 2 is a simplified display window 200 from custom client software 145. Display window 200 includes various control areas such as control area 210 and 220. Control areas typically include control mechanisms such as links and buttons to control the functions of custom client software 145. For example, if custom client software 145 is an email client, control areas 220 and 230 may include control mechanisms for reading email, composing new email, forwarding an email, or responding to an email. Display window 200 also includes a data list area 240 and a data record display area 250. Data list area 240 includes data identifiers 240-1, 240-2, … 240-7. The actual number of data identifiers may be much larger than seven. Typically, if more there are more data identifiers than can be shown on in data list area 240, data list area 240 has some mechanism, such as a scroll bar, to selectively show groups of data identifier. Data identifiers differ depending on the specific application. For example, if custom client software 145 is an email client, each data identifier indicates an email message. For email software, data identifier 240-1 may include information such as the sender, date, and subject of a corresponding email. Furthermore, data identifier 240-1 may contain additional information that is

not displayed. Data list area 240 also includes a current data identifier marker 245. Current data identifier marker 245 indicates the data identifier, which corresponds to the current data record being displayed in data record display area 250. Some embodiments of data list area 240 also include status indicators (not shown) about the corresponding data record. For example, in an email application, status indicators may indicate whether the email is read, unread, or whether a reply to the email has already been sent.

[0010] Data record display area 250 displays the current data record corresponding to the current data identifier. In Fig. 2, data record display area 250 displays data record 250-2, which corresponds to data identifier 240-2, which is the current data identifier as indicated by current data identifier marker 245. In addition, data record display area 250 includes control buttons such as next button 251 and previous button 252. Next button 251 causes the data identifier following the current data identifier to become the current data identifier. In addition, the data record following the current data record becomes the current data record. Furthermore, the new current data record is displayed in data record display area 250 and current data identifier marker 245 is updated to point to the new current data identifier. For example, in Fig. 2 clicking next button 251 causes data identifier 240-3 to become the current data identifier; data record 250-3 (not shown) is displayed in data record display area 250, and current data identifier marker 245 is moved to point to data identifier 240-3.

[0011] Conversely, previous button 252 causes the data identifier preceding the current data identifier to become the current data identifier. In addition, the data record preceding the current data record becomes the current data record. Furthermore, the new current data record is displayed in data

record display area 250 and current data identifier marker 245 is updated to point to the new current data identifier. For example, in Fig. 2 clicking previous button 252 causes data identifier 240-1 to become the current data identifier; data record 250-1 (not shown) is displayed in data record display area 250, and current data identifier marker 245 is moved to point to data identifier 240-1.

[0012] Many computer users require access to data records from server software 115 from different computers. For example, an employee may need to access data records from server software 115 while traveling. For example, in Fig. 1, client computer 160 may be located in a different city than server computer 110 and client computer 140. Access to server computer 110 from client computer 160 is generally limited by the bandwidth of wide area network 150. Furthermore, the specific software such as custom client software 145 may not be available on client computer 160. Therefore, server software 115 is often configured to support use of generic client software 165. In general, generic client software 165 contacts server software 115 and receives computer instructions, which configures generic client software 165 to operate with server software 115 using industry standard protocols such as HTTP and JavaScript.

[0013] However, several issues cause difficulties in replicating the features of custom client software 145 using generic client software 165. One issue is the slow speed of wide area network 150 compared to local area network 120. For example, in most implementations of custom client software 145, all the data identifiers are transmitted to custom client software 145. Thus, custom client software 145 allows a user to easily scan through portions of the data identifiers to locate a desired data record. However, the latency caused by transferring

a large list of data identifiers using wide area network 150 may be unacceptable.

[0014] Another issue is due to the static nature of generic client software 165. Specifically, generic client software 165 generally requests specific data pages, such as a web page, from server software 115 using a uniform resource locater (URL). Server software 115 processes the request from generic client software 165 and sends a data page for generic client software to display. The data pages may include links (embedded URLs), which can be selected to request another data page. Thus, for example some web based email systems display a subset of the list of email message headers as links, which can be selected to display a corresponding email message in place of the email message headers. However, conventional configurations of generic client software 165 can not replicate the dual display areas typical of custom client software 145. Hence, there is a need for a method for configuring generic client software to provide the features of custom client software using industry standard protocols.

SUMMARY

[0015] Accordingly, generic client software, such as web browsers, is configured to allow different display frames to be synchronized in accordance with one embodiment of the present invention. The synchronization provided by the present invention allows common custom client software features, such as a current data identifier marker and synchronized data list viewing to be implemented. Specifically, in one embodiment of the present invention a data display system is implemented by configuring generic client software. The data display system includes a data display frame and a data list frame. The data display frame is configured to display a current data record. The data list frame is configured to display a set of data identifiers and a current

data identifier marker. The current data identifier marker indicates the current data identifier which corresponds to the current data record. The data display system can also include a parent frame that contains both the data display frame and the data list frame, as well as, variables and command scripts for viewing and manipulating the data records.

[0016] For example, some embodiments of the present invention includes a next command script. The next command script is generally activated by using a next button that is displayed in either the data list frame or the data display frame. Similarly, most embodiments of the present invention would also include a previous button to activate a previous command script. The next command script is configured to request a new current data record and to update the current data identifier marker. Furthermore, in some embodiments, the command script is also configured to request additional data identifiers when the current data record corresponds to the last data identifier in the data list frame.

[0017] Some embodiments of the present invention also includes status markers for the data identifiers. For example, in one embodiment of the present invention the data display system is an email client implemented on a web browser. In the email client, the data identifiers correspond to data records that are email messages. Status markers in the data list frame can indicate whether the email message has been read, forwarded, or deleted.

[0018] The present invention will be more fully understood in view of the following description and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0019] Fig. 1 is a conventional computer network.

[0020] Fig. 2 is a simplified display window of a conventional custom client software.

[0021] Fig. 3(a) is a simplified display window of generic client software configured in accordance with one embodiment of the present invention.

[0022] Fig. 3(b) is a data display frame in accordance with one embodiment of the present invention

[0023] Fig. 4(a) is a simplified display window of generic client software configured in accordance with one embodiment of the present invention.

[0024] Fig. 4(b) is a simplified display window of generic client software configured in accordance with one embodiment of the present invention.

[0025] Fig. 5 is a diagram of a parent frame in accordance with one embodiment of the present invention.

[0026] Fig. 6 is a flow diagram of a command script in accordance with one embodiment of the present invention.

DETAILED DESCRIPTION

[0027] Fig. 3(a) is a display window 300 from generic client software 165 configured in accordance with one embodiment of the present invention. Display window 300 is separated using frames, which segments display window 300 into distinct sections. In general, each frame can be treated as a separate data page. Display window 300 includes a control frame 310, a control frame 320, a parent frame 330, data list frame 340, and data record display frame 350. Parent frame 330, which is drawn with dashed lines, is generally not visible to a user viewing display window 300. For clarity, elements drawn with dashed lines in the figures are generally not visible to a user. Display window 300 is made to resemble display window 200 so that a user of custom client software 145 can easily adapt to using generic client software 165 to access server software 115. Thus, control frames 310 and 320 include similar control mechanisms as control areas

210 and 220. Similarly, Data list frame 340 displays data identifiers 340-1 through 340-7 as well as a current data identifier marker 345. The data identifiers in data list frame 340 are a subset of the data identifiers available from server software 115. Different embodiments of the present invention can display greater or fewer numbers of data identifiers in data list frame 340. Because of the bandwidth limitations of wide area network 150, subsets of the data identifiers are sent to generic client software 165 when needed to display. Although some embodiments of the present invention may send more data identifiers than can be currently displayed. Some embodiments of the present invention may also include a status indicator for each data identifier. For example, in Fig. 3(b) data list frame 340 also includes status indicators 340-1-S to 340-7-S, which indicates the status of data record 350-1 to 350-7, respectively. In the embodiment of Fig. 3(b) status indicators 340-1-S to 340-7-S, can use different colors to indicate different states. Alternatively, some embodiments of the present invention may use different icons to indicate different states. Still other embodiments of the present invention may use highlighting, bolding, or different colors with the data identifiers to indicate different states.

[0028] Data record display frame 350 has a next button 351, a previous button 352, and displays the current data record, which corresponds with the current data identifier. In Fig. 3(a), data display frame 350 displays data record 350-2, which corresponds to data identifier 340-2, which is the current data identifier as indicated by current data identifier marker 345. Next button 351 and previous button 352 perform the same functions as next button 251 and previous button 252, respectively. However, next button 351 and previous button 352 are implemented using industry standard protocols such as JavaScript.

[0029] As explained above, each frame can be considered a separate data page from server software 115. When the contents of a frame changes, generic client software 165 requests a new data page from server software 115. Thus, synchronization between two different frames is difficult especially if both frames must change at the same time. For example, in Fig 4(a) current data identifier 345 points to data identifier 340-7. Accordingly, data record display frame 350 displays data record 350-7. If a user hits next button 351, both data record display frame 350 an data list frame 340 must be updated. Specifically, as shown in Fig. 4(b), after hitting next button 351, data list frame 340 should lists data identifiers 340-8 to 340-14 and current data identifier marker 345 should point to data identifier 340-8. In addition, data record display frame 350 should display data record 350-8. However, conventional configurations of generic client software 165 are unable to provide this synchronization between frames.

[0030] A major reason synchronization has not been achieved is due to the latency, i.e. delay, of the network connection between server computer 110 and client computer 160. For example, after server software 115 dispatches a new data page to generic client software 165, server software 115 does not actually know when the dispatched page will arrive and be processed by generic client software 165. Thus, server software 115 is generally programmed to assume all new information from generic client software 165 is sent after receipt of the data page dispatched by server software 115. However, the data page sent by server software 115 may be delayed due to network congestion. Thus, generic client software 165, under control of an impatient user, may issue additional requests while using a data page that should have been replaced by the data page sent by server software 115. Thus, the information or request from generic client software 165 would

likely be misinterpreted by server software 115. Consequently, server software 115 and generic client software 165 are not properly synchronized. This lack of synchronization can cause unintended and erroneous behavior in both server software 115 and generic client software 165.

[0031] Another problem causing loss of synchronization is caused by the transfer protocol used on wider area network 150 and local area network 120. Specifically, communications over computer networks generally uses packet based protocols, such as TCP/IP. In packet based protocols, data transmitted over the network are broken up into small packets. The packets, which are sent individually over the computer network, can arrive at the destination out of order but are encoded so that the packets can be reassembled properly by the networking software (not shown) on each computer. When only using local area network 120, even out of order packets will arrive quickly enough for custom client software 145 to seamlessly update multiple areas of display window 200. However, if the packets must travel over wide area network 150, the packets may arrive too slowly for generic client software 165 to update multiple frames without errors.

[0032] The present invention solves the various issues described above by using parent frame 330. Both data list frame 340 and data record display frame 350 are children frames of parent frame 330. As shown in Fig. 5, in addition to data list frame 340 and data record display frame 350, parent frame 330 includes command scripts 510, command variables 520, a command queue 530, data list frame lock 540, and data record display frame lock 550. Command scripts 510, command variables 520, command queue 530, data list frame lock 540, and data record display frame lock 550 are generally are not visible. Command scripts 510 includes the implementation of next button 351 and previous button 352. Various other command scripts are included

depending on the specific application implemented on generic client software 165. For example, if generic client software 165 is configured for email, command scripts 510 may also include a delete command script, a reply command script, and a forward command script. Corresponding buttons or links for these command scripts would also be included in data record display frame 350. The computer program listing appendix includes command scripts using JavaScript for an embodiment of the present invention which configures generic client 165 as an email client.

[0033] Command variables 520 includes any variables needed by command scripts 510. Furthermore, command variables 520 may also include persistent variables required by any command scripts in data list frame 340 or data record display frame 350. Thus, from the perspective of data record display frame 350 and data list frame 340 command variables 520 are similar to global variables in normal programming languages. Command variables 520 includes a current data identifier variable C_IDEN (not shown) which identifies the current data identifier. The specific variables included in command variables 520 depends on the application implemented by configuring generic client software 165.

[0034] Command queue 530 stores commands issued by the user of generic client software 165. For example, a user may clock on next button 351 several times. However, due to the limited bandwidth and latency of wide area network 150, parts of the data pages necessary for the command scripts associated with next button 351 may not be available. Thus, the execution of the command scripts are queued in command queue 530 until the necessary data is available.

[0035] Data list frame lock 540 and data record display frame lock 550 are indicators of whether data list frame 340 and data record display frame 350, respectively, contain valid data pages. Specifically, in one embodiment of the present invention, data

list frame lock 540 enters a lock state when data list frame 340 requests a new data page from servers software 115. After the new data page is fully received, data list frame lock 540 enters an unlock state. For embodiments of the present invention using HTML, the ONLOAD command can be used to change the state of data list frame lock 540 when a data page is loaded into data list frame 340. When a command script needs to access the data page of data list frame 340 while data list frame lock 540 is in the lock state, the command script is placed in command queue 530. Similarly, data record display frame lock 550 enters a lock state when data record display frame 350 requires a new data page from servers software 115. After the new data page is fully received, data record display frame lock 540 enters an unlock state.

[0036] Fig. 6 is a flow diagram illustrating the NEXT command script associated with next button 351 for one embodiment of the present invention. Upon execution, the NEXT command script checks the status of data list frame lock 540 and data record display frame lock 550 in a check lock step 610. If either data list frame lock 540 or data record display frame lock 550 is in the lock state, the NEXT command is stored in command queue 530 during a store in command queue step 620. Execution of commands in command queue 530 are triggered using the HTTP ONLOAD command when data list frame 340 or data record display frame 350 are loaded. If both data list frame lock 540 and data record display frame lock 550 are in the unlock state, the NEXT command script requests the next data record from server software 115 in request next data record step 630. In response to the request, server software 115 sends a new data page for data record display frame 350 containing the next data record. The current data identifier variable is incremented in increment current data identifier variable step 640.

[0037] In check data list frame step 650, a determination is made whether the new current identifier is already listed in data list frame 340. If the new current identifier is already listed in data list frame 340, current data identifier marker 345 updated to the new current data identifier and any status indicators are also updated in update status and current marker step 670. If the new current data identifier is not listed in data list frame 340, the NEXT command script requests server software 115 to send a new data page for data list frame 340 which includes the new current data identifier in request new group of data identifiers step 660. In some embodiments of the present invention, check data list frame step 650 is accomplished by first determining if the current data identifier is the last data identifier listed in data list frame 340.

[0038] In the embodiment of Fig. 6, the NEXT command script actually ends after request new group of data identifiers step 660. However, the data page sent by server software 115 in response to the request issued by the NEXT command script includes a ONLOAD command to update the status and current data identifier marker. Thus, as shown in Fig. 6, after the new data page is loaded into data list frame 340, the ONLOAD command is used in ONLOAD of data list frame step 680. Then, in update status and current marker step 690 the status fields and current data identifier 345 in data list frame 340 are updated.

[0039] A previous command script associated with previous button 352 functions in a similar fashion. A set of command scripts for use with an embodiment of the present invention configured for an email application is included in the computer program listing appendix.

[0040] In the various embodiments of this invention, novel structures and methods have been described for synchronizing multiple frames of a generic client software, such as a web

browser. Synchronization between frames allows common features found on custom client software on local area networks to be implemented on generic client software over wide area networks. For example, one embodiment of the present invention uses a data list frame and a data display frame. The data list frame displays a list of data identifiers and maintains a current data identifier marker on the current data identifier which corresponds to a current data record in the data display frame. The various embodiments of the structures and methods of this invention that are described above are illustrative only of the principles of this invention and are not intended to limit the scope of the invention to the particular embodiments described. For example, in view of this disclosure, those skilled in the art can define other frames, command scripts, command variables, data identifiers, data records, generic client software, server software, and so forth, and use these alternative features to create a method or system according to the principles of this invention. Thus, the invention is limited only by the following claims.